

Adaptive Shadow Maps

Randima Fernando
Sebastian Fernandez
Kavita Bala
Donald P. Greenberg

Shadow Maps

- **Introduced by Lance Williams (SIGGRAPH 1978)**
- **Advantages**
 - Flexible
 - Fast
 - Easy to implement
- **One major disadvantage: aliasing**
- **Our solution: the Adaptive Shadow Map**

Related Work

- Casting Curved Shadows on Curved Surfaces (Williams, 1978)
- Algorithms for Antialiased Cast Shadows (Hourcade and Nicolas, 1985)
- The Light Buffer: a Shadow Testing Accelerator. (Haines and Greenberg, 1986)
- Rendering Antialiased Shadows with Depth Maps (Reeves et al., 1987)
- Hierarchical Z-Buffer Visibility (Greene and Kass, 1993)
- Hierarchical Rendering of Trees from Precomputed Multi-Layer Z-Buffers (Max, 1996)
- Deep Shadow Maps (Lokovic and Veach, 2000)
- ...

A Comparison



**Conventional Shadow
Map**

(2048 x 2048 pixels)

16 MB Memory

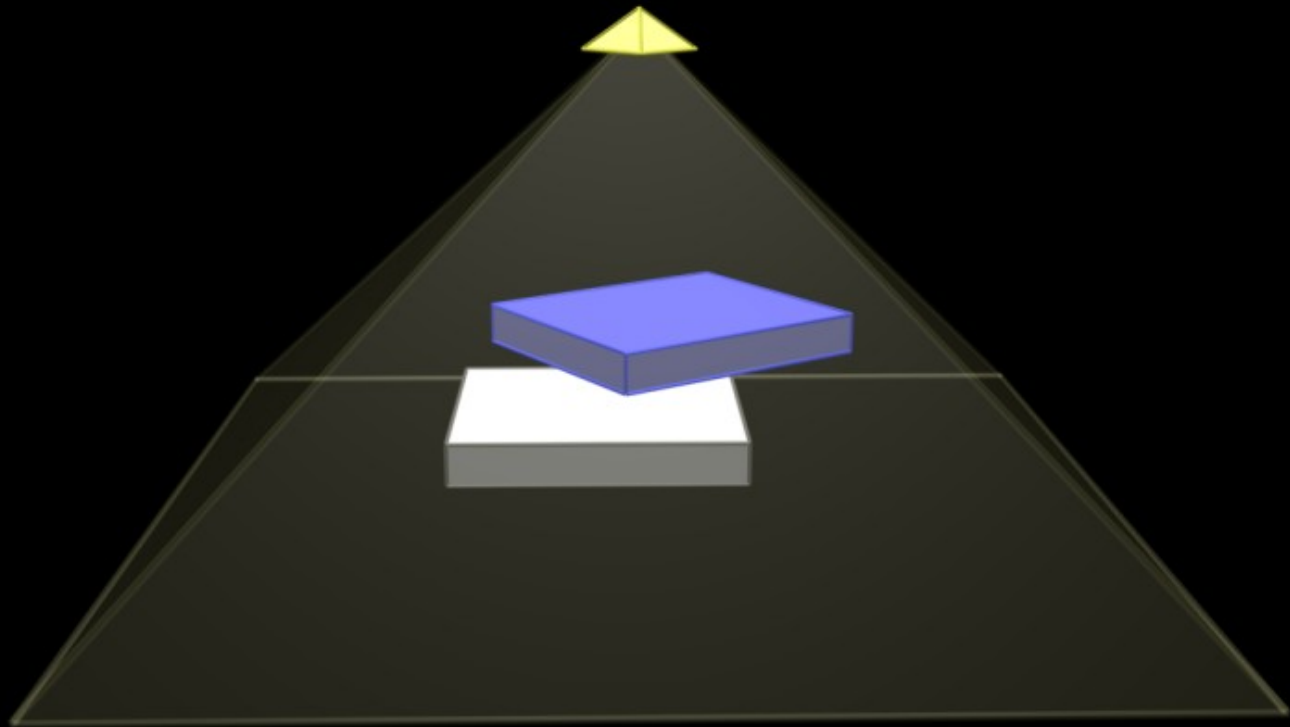


**Adaptive Shadow
Map**

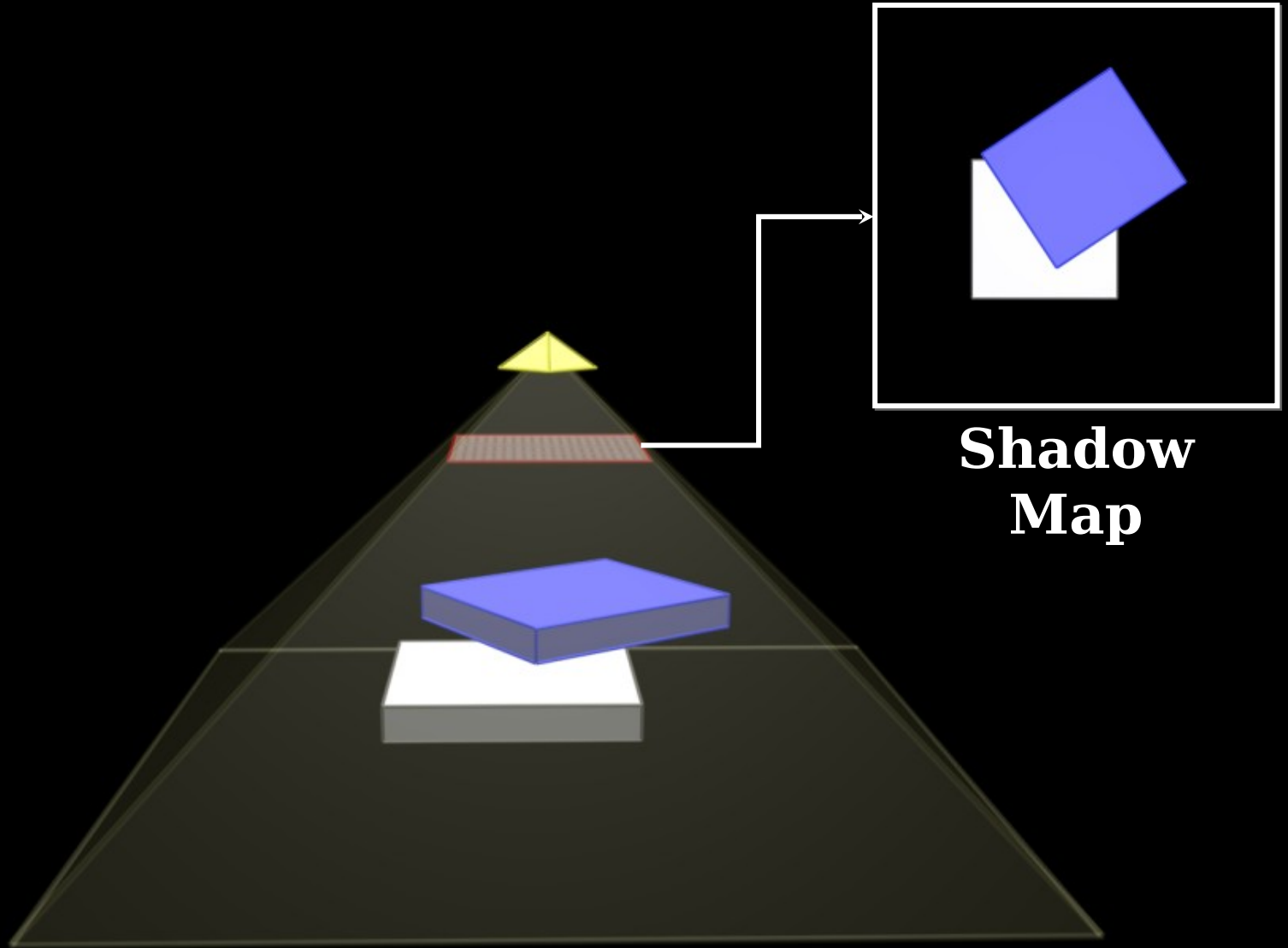
(Variable Resolution)

16 MB Memory

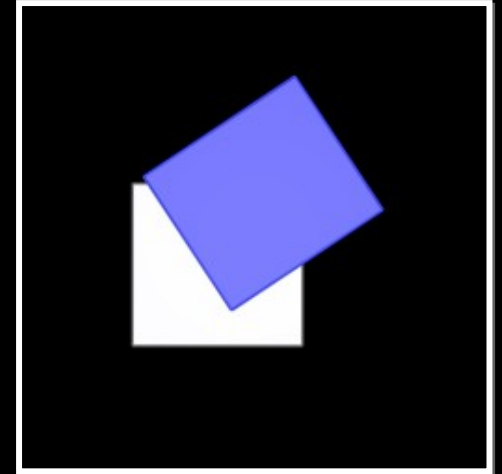
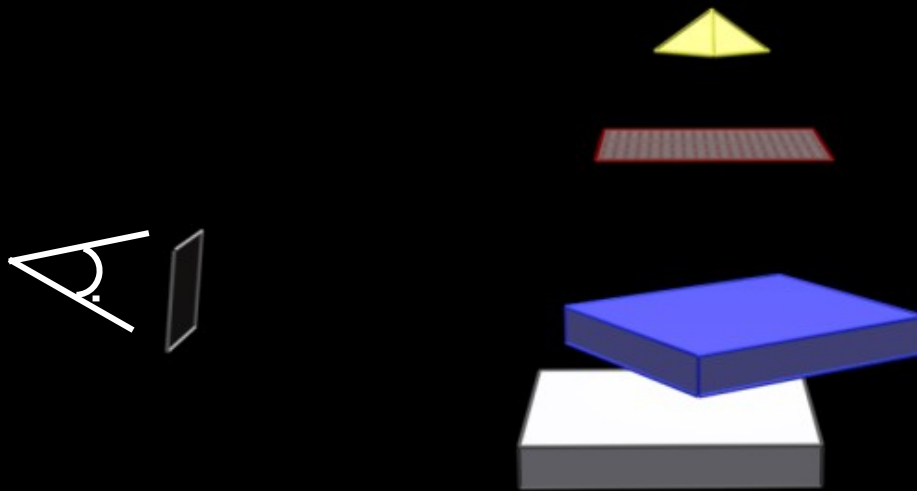
How Shadow Maps Work



How Shadow Maps Work

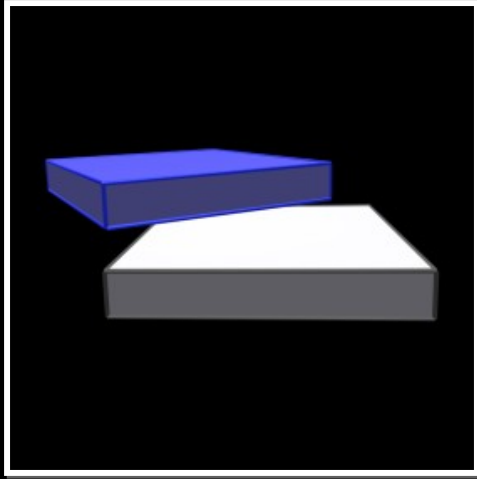


How Shadow Maps Work

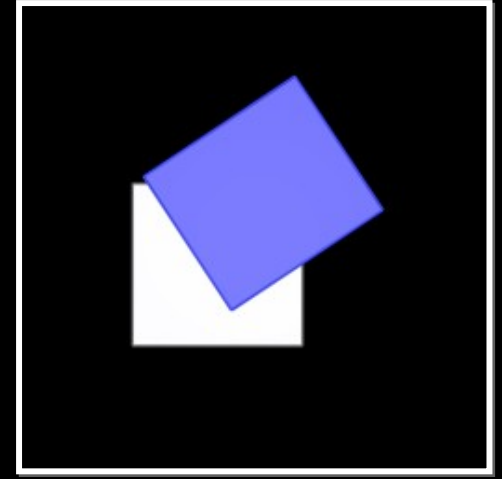
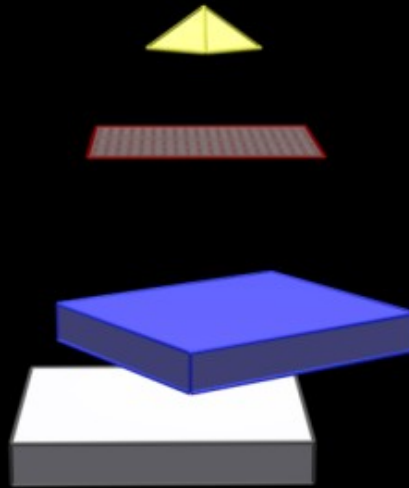


**Shadow
Map**

How Shadow Maps Work

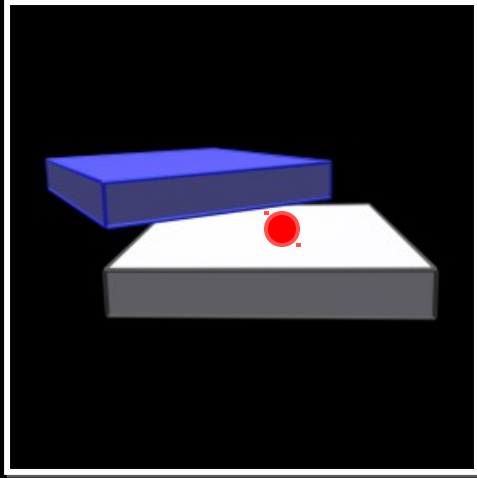


Eye View

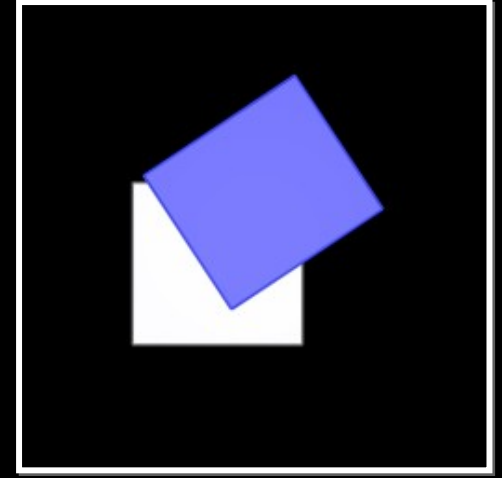
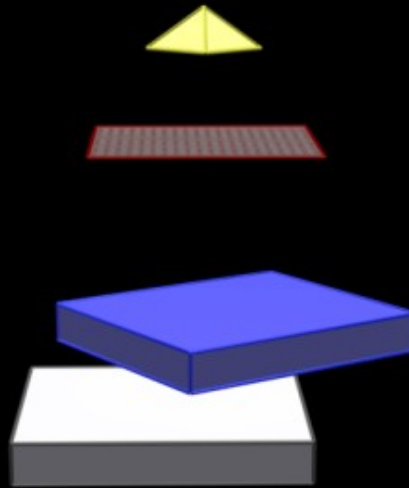


Shadow Map

How Shadow Maps Work

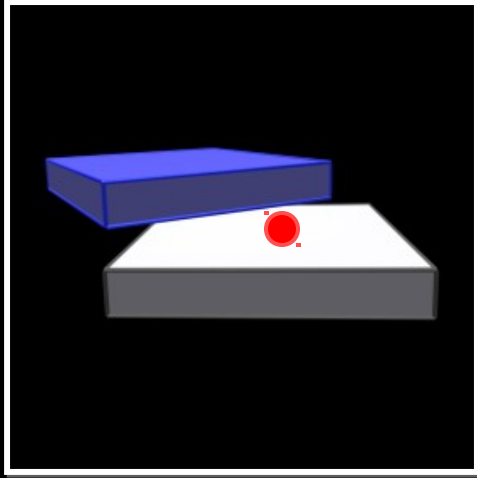


Eye View

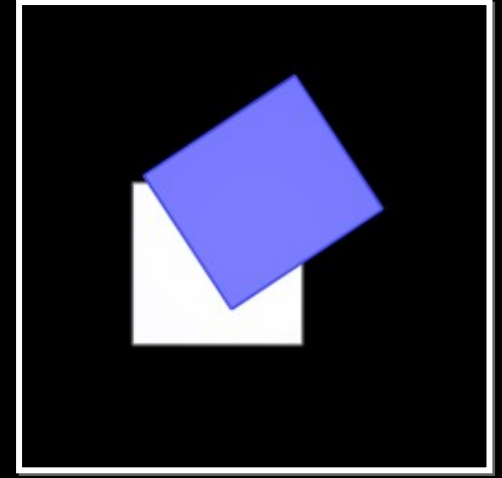


Shadow Map

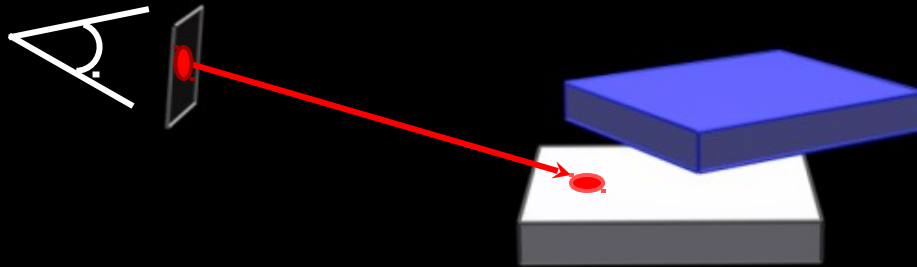
How Shadow Maps Work



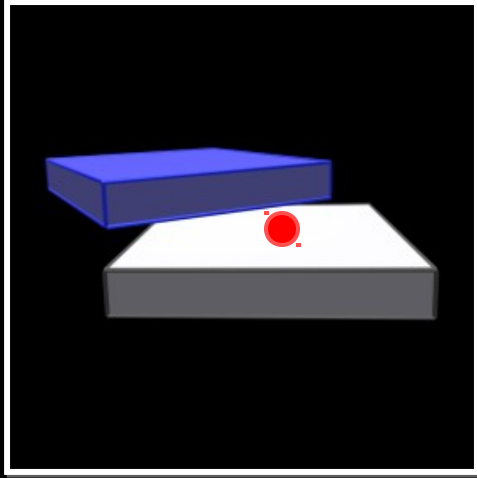
Eye View



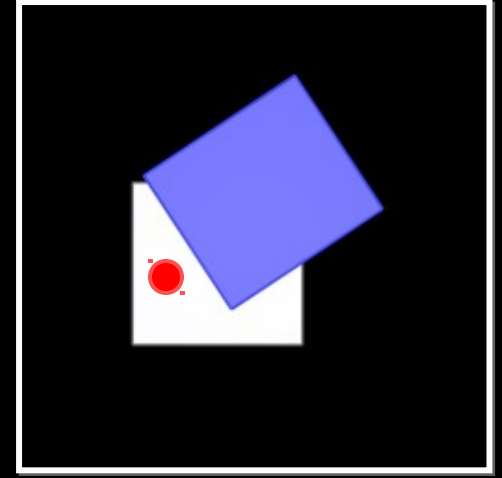
Shadow Map



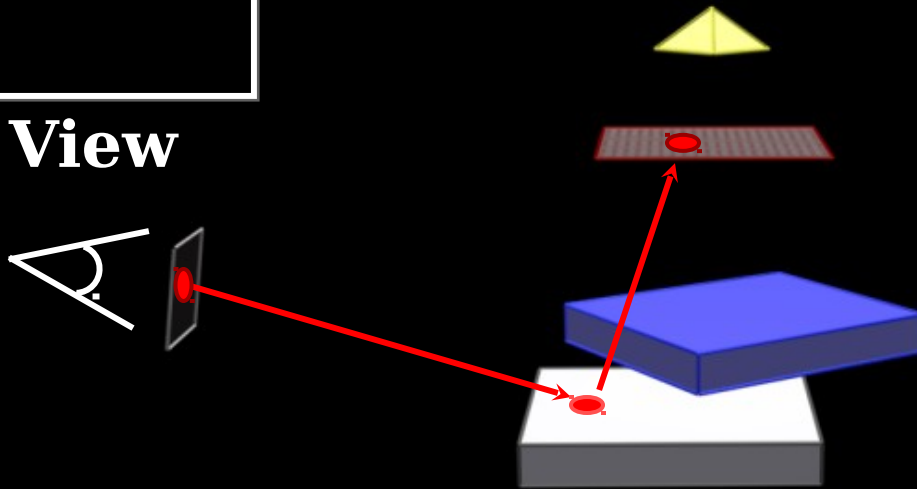
How Shadow Maps Work



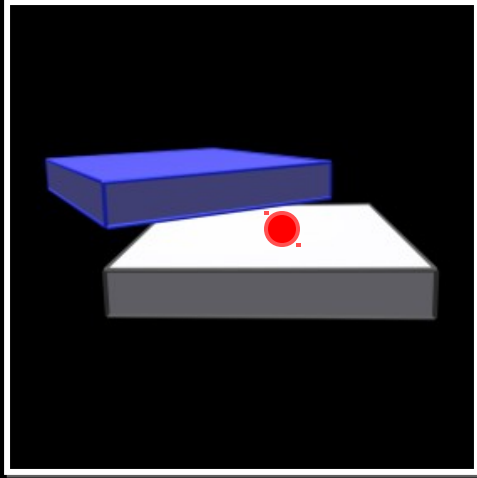
Eye View



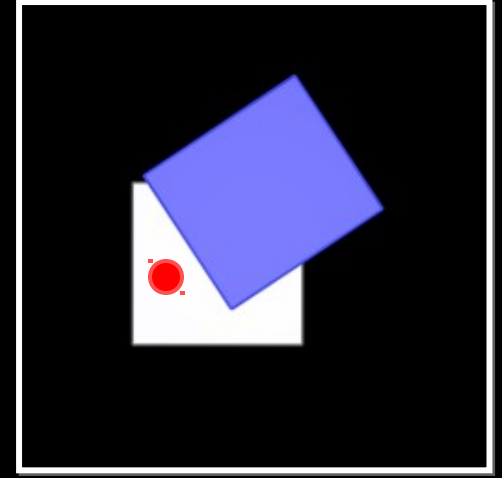
Shadow Map



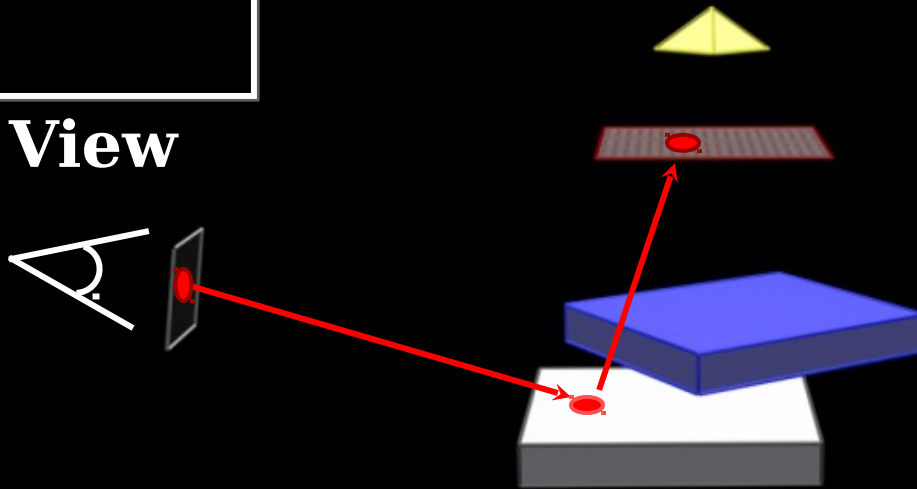
How Shadow Maps Work



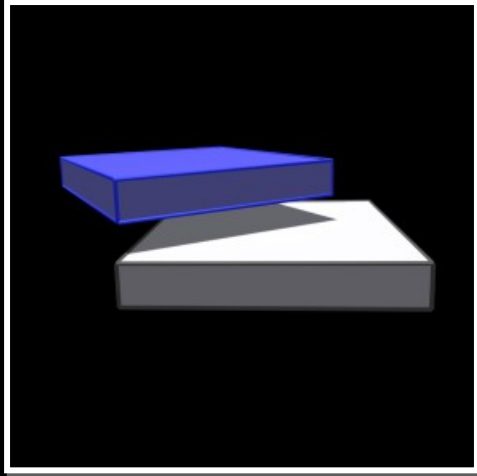
Eye View



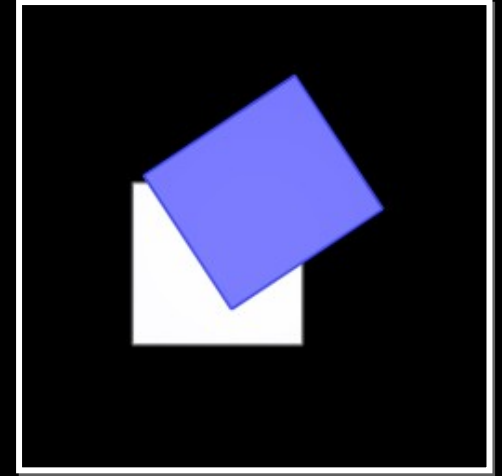
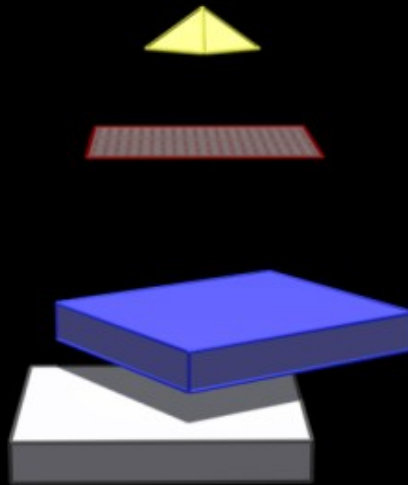
Shadow Map



How Shadow Maps Work

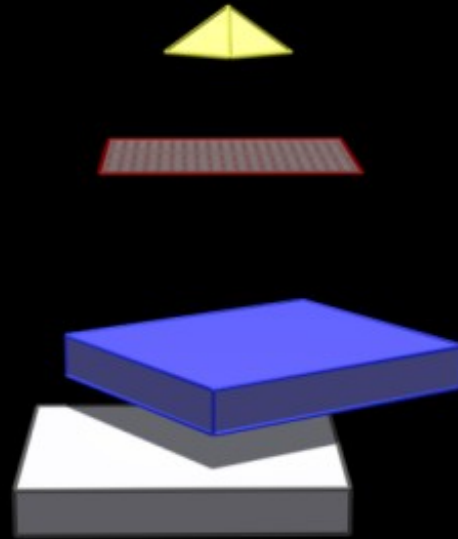


Eye View

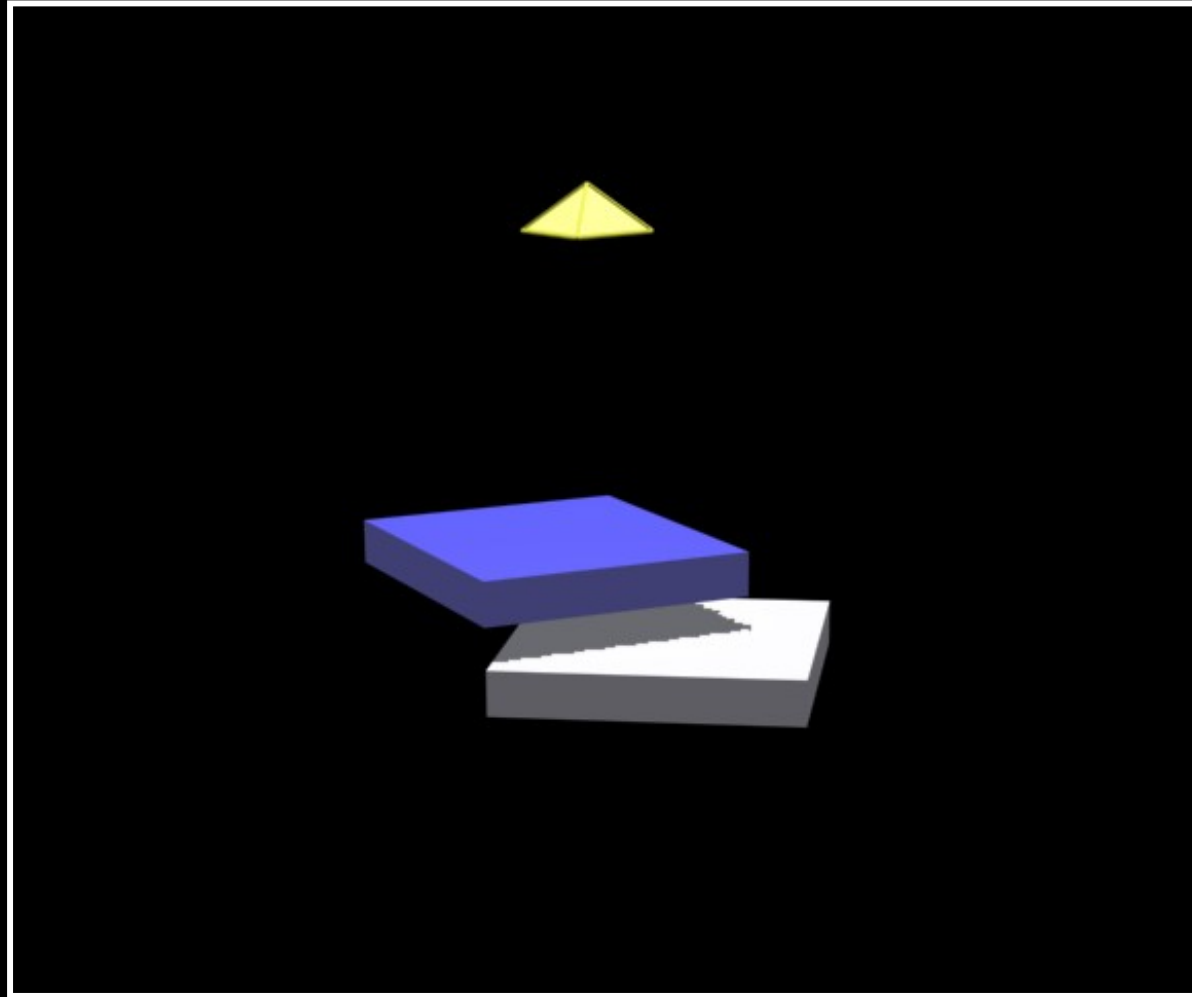


Shadow Map

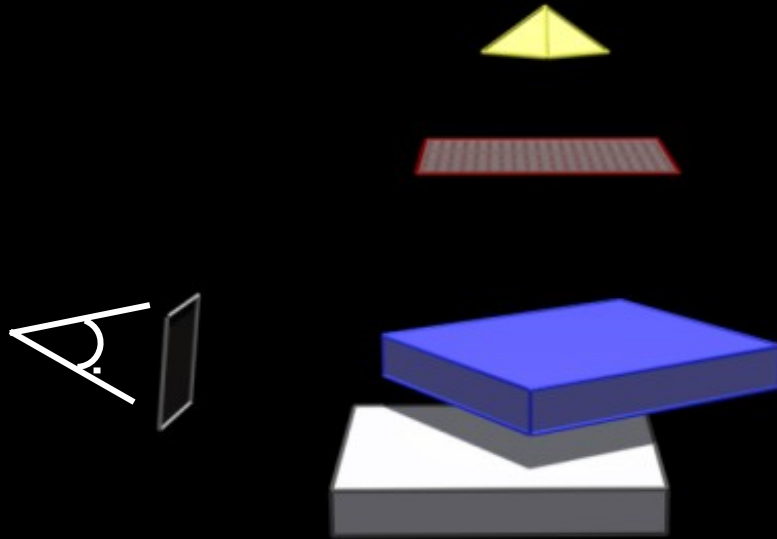
Aliasing (Distant)



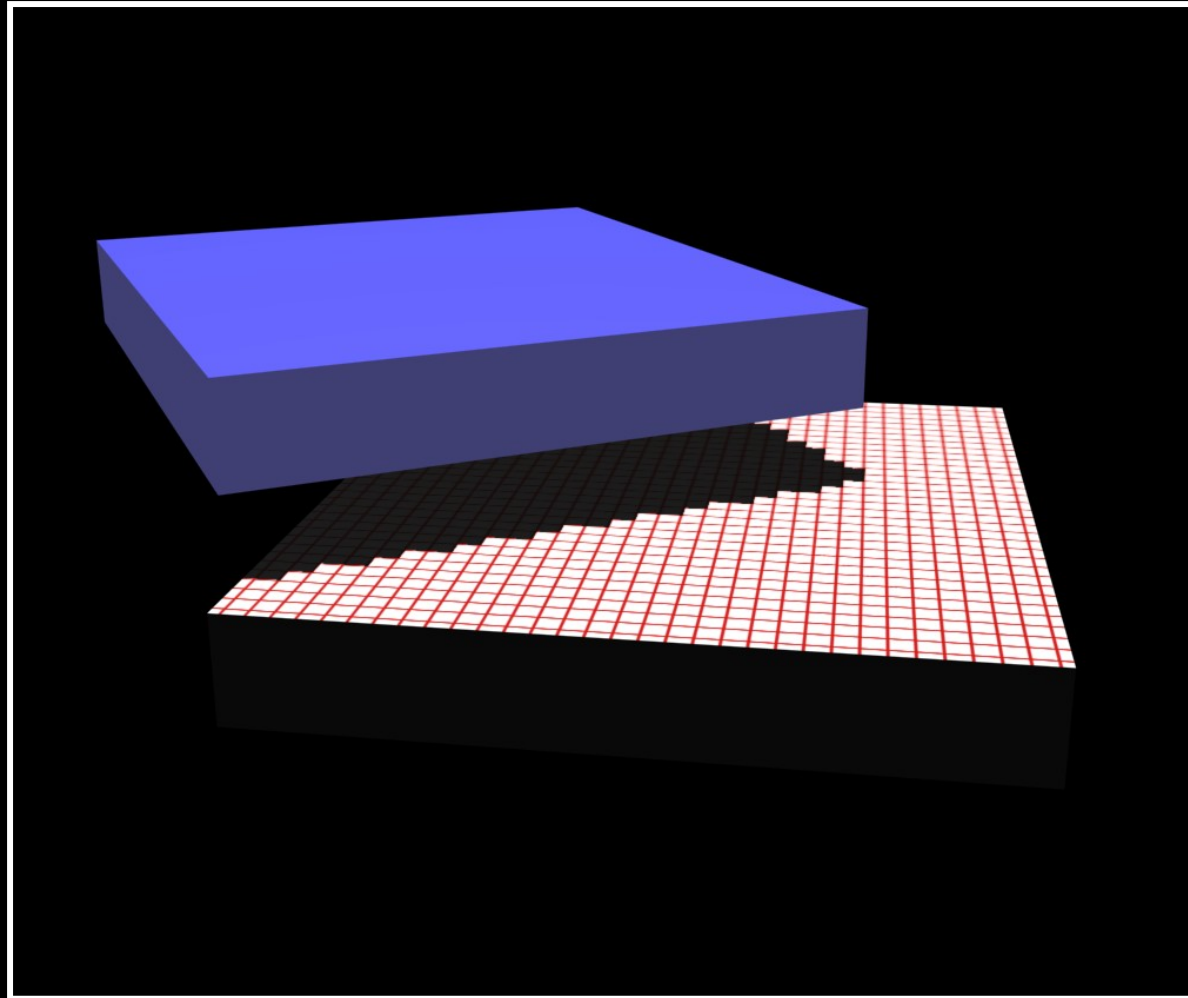
Aliasing in Eye View (Distant)



Aliasing (Distant)



Aliasing in Eye View (Close)



Motivation

- **Shadow maps require too much tweaking**
 - Where to place light?
 - What resolution to use?
- **Goals:**
 - Address the aliasing problem
 - No user intervention
 - Interactive frame rate

Adaptive Shadow Maps

- **Idea:**

- Refine shadow map on the fly

- **Goal:**

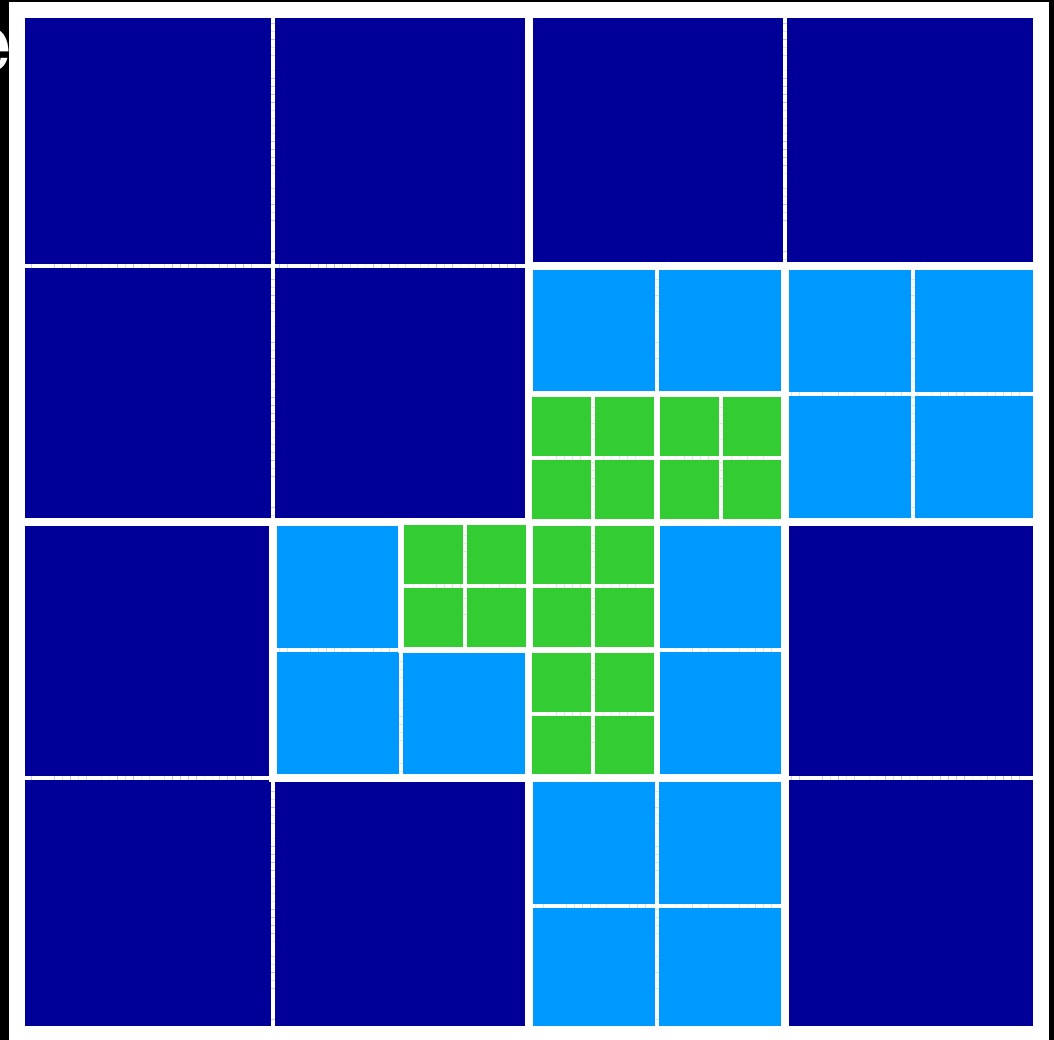
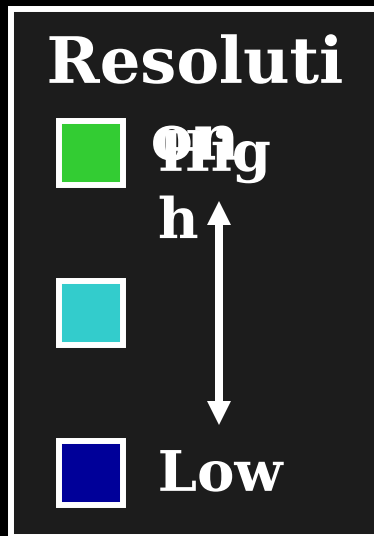
- Shade each eye pixel with a different shadow map pixel

- **Implementation:**

- Use hierarchical structure for shadow map
- Create/delete pieces of shadow map as needed
- Exploit fast rendering and frame buffer read-backs

Adaptive Shadow Map Data Structure

- Simple 2D tree

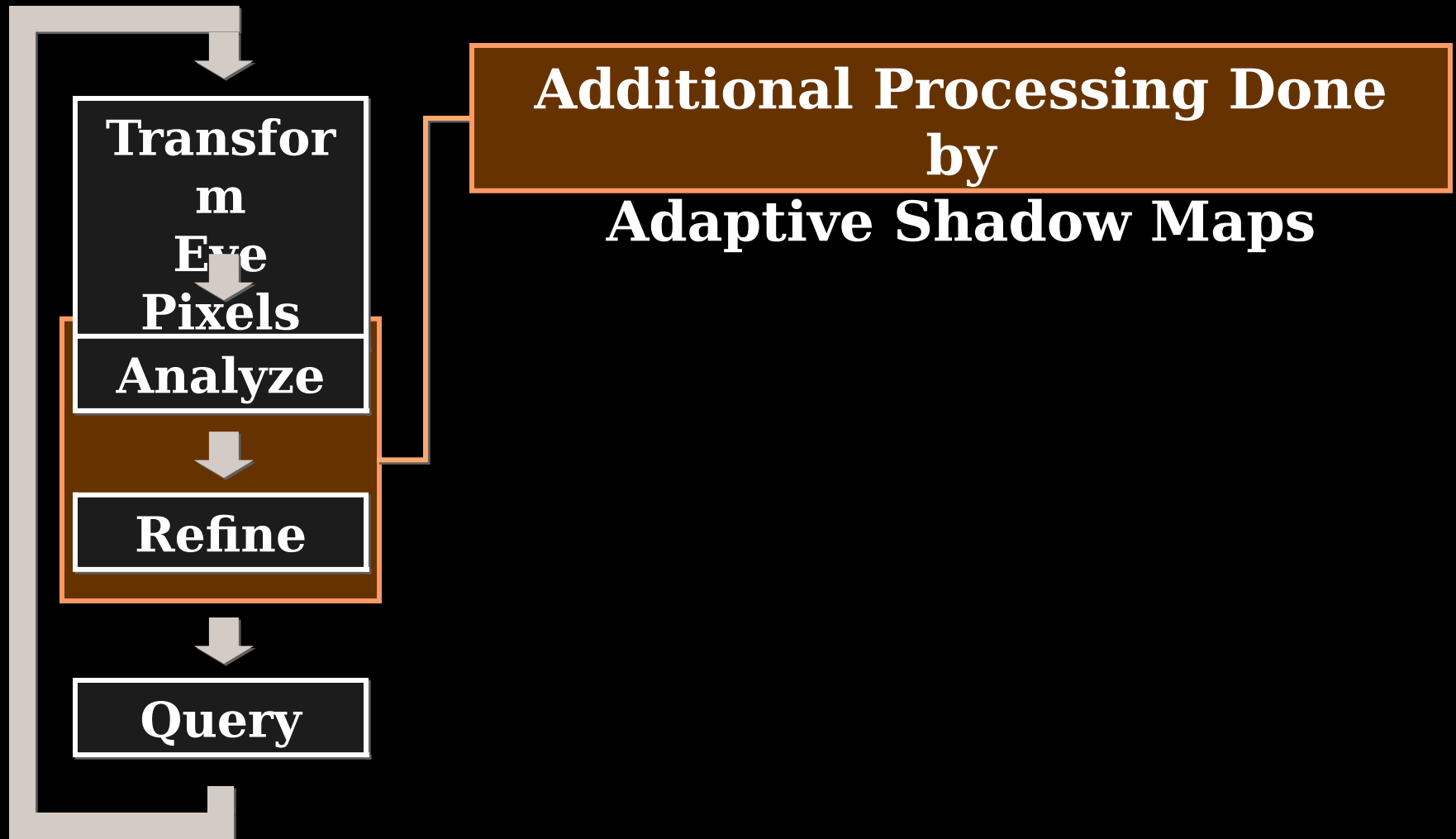


Adaptive Shadow Map Characteristics

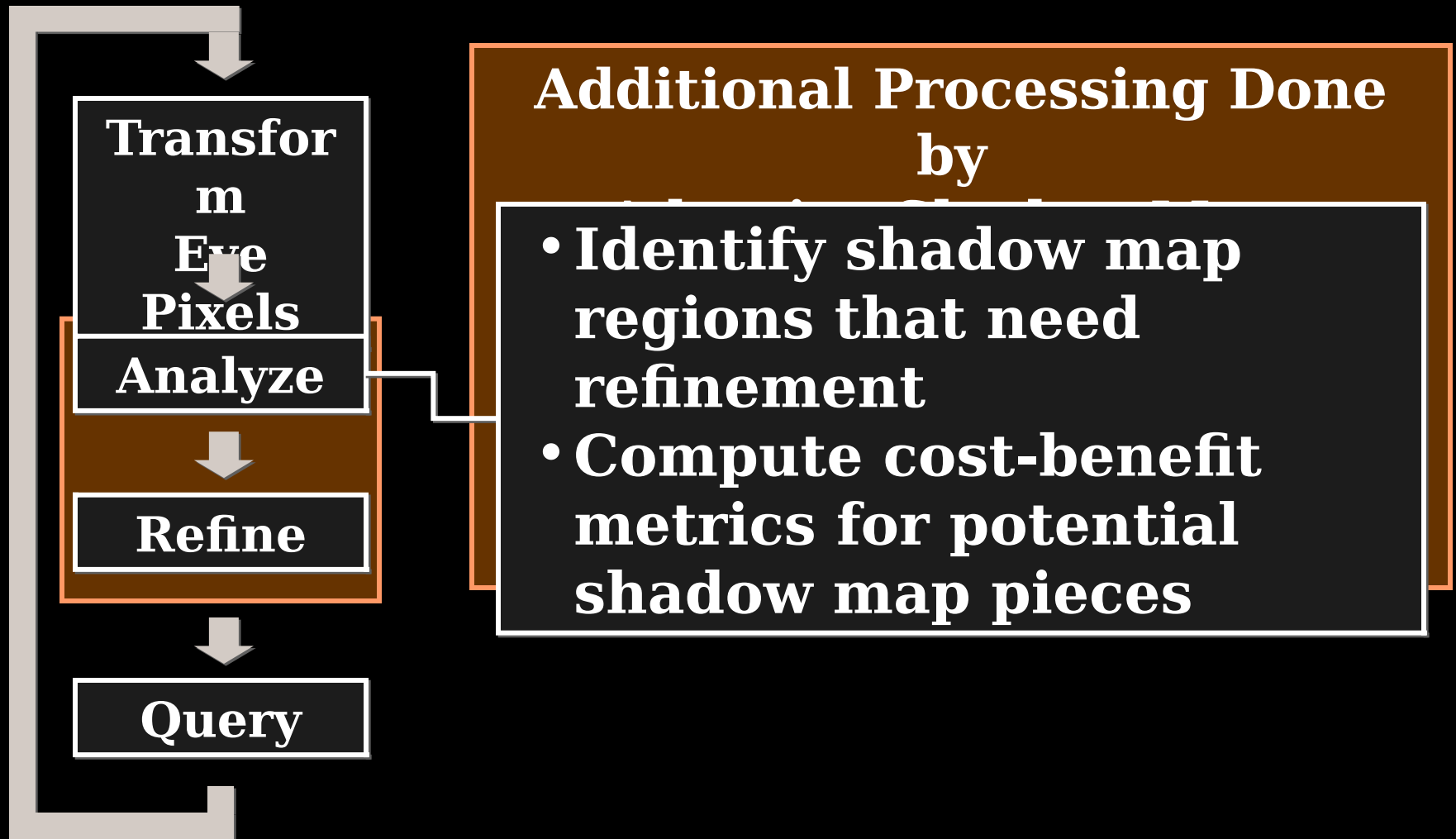
- View-driven
- Constrained memory usage
- Progressive
- No manual intervention
- Usage same as conventional shadow map:



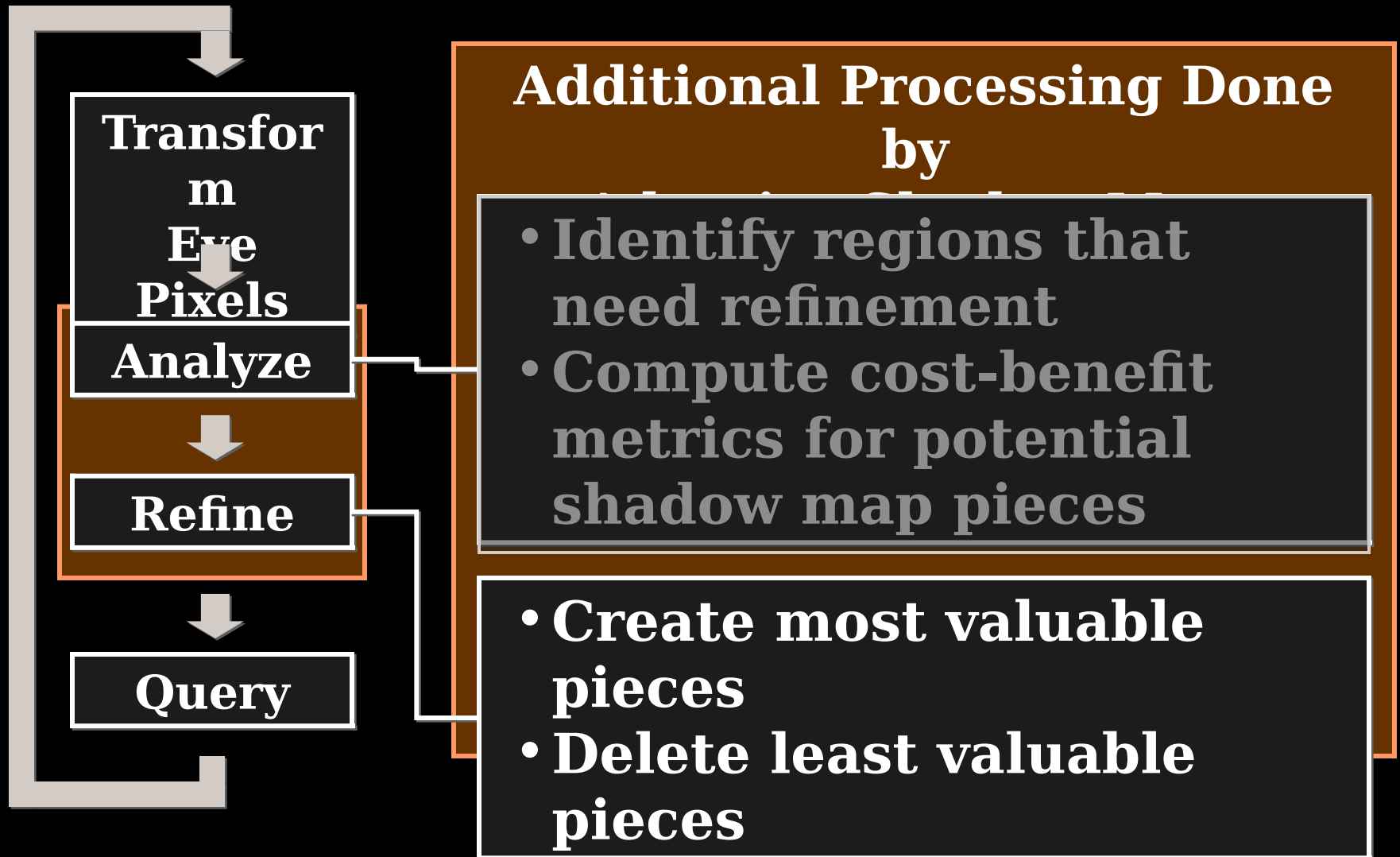
Process Flow



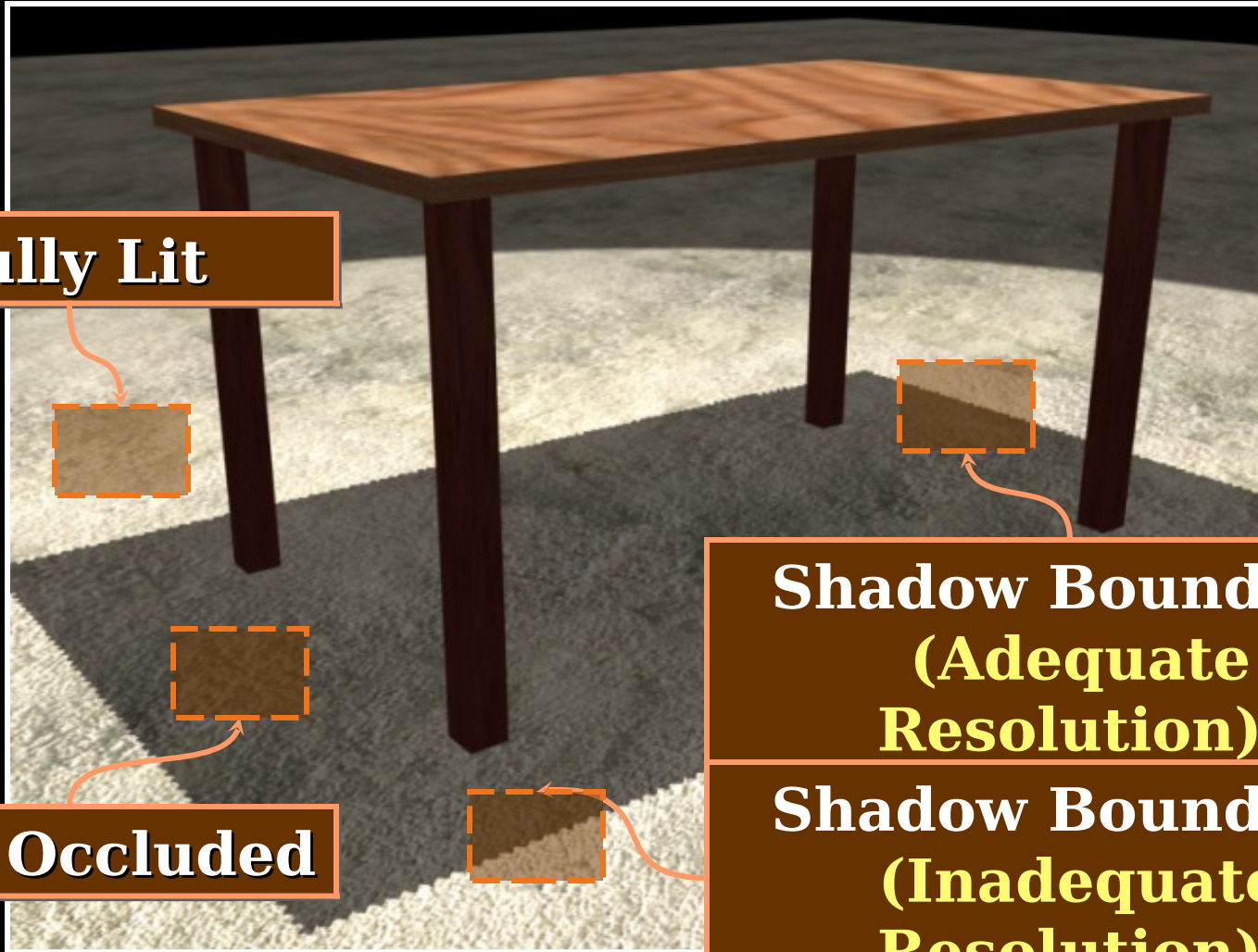
Process Flow



Process Flow



Where to Refine?



Fully Lit

Fully Occluded

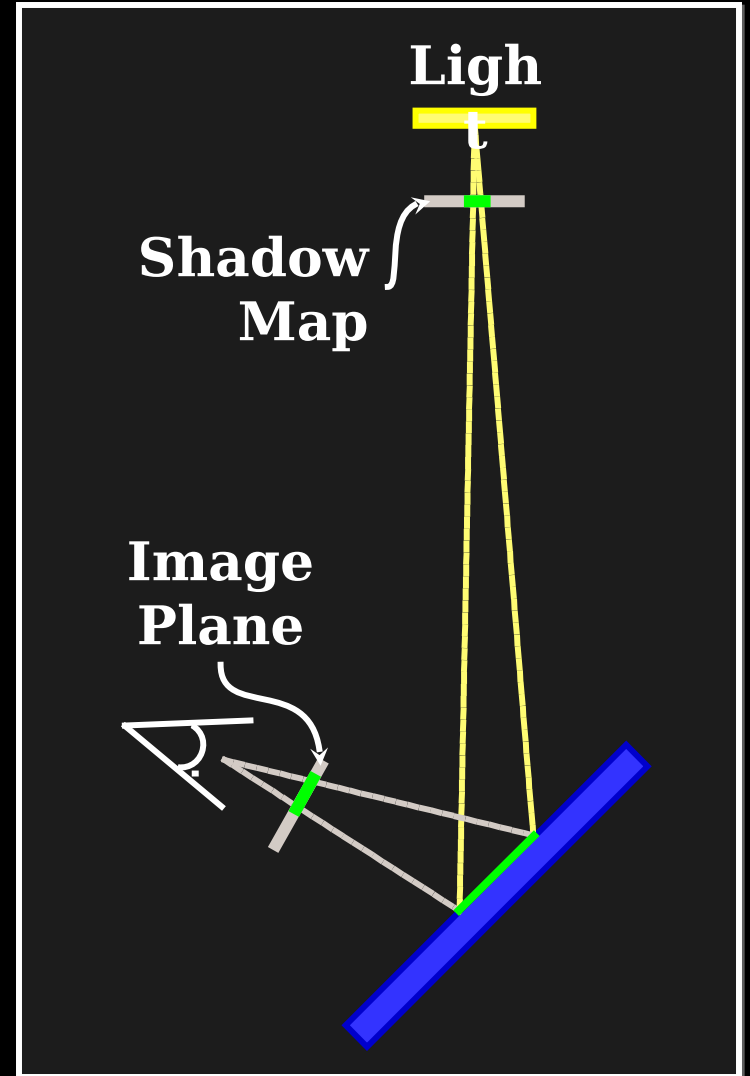
**Shadow Boundary
(Adequate
Resolution)**

**Shadow Boundary
(Inadequate
Resolution)**

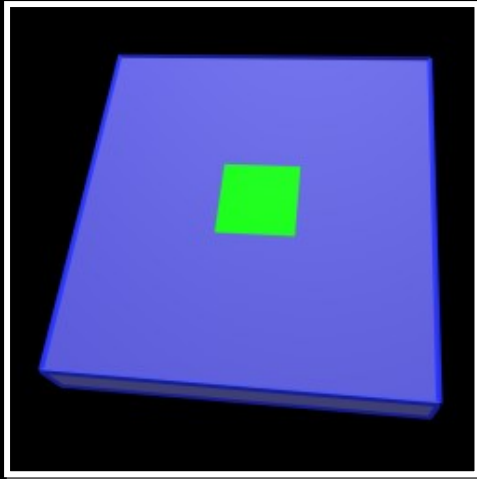
Resolution-Mismatched Edge Pixels

- **Eye view pixels that:**
 - Lie on shadow boundaries, *and*
 - Have a larger projected pixel size in the eye view than in the shadow map
- **Don't want to refine:**
 - Fully lit / fully occluded regions
 - Shadow boundary regions of adequate resolution

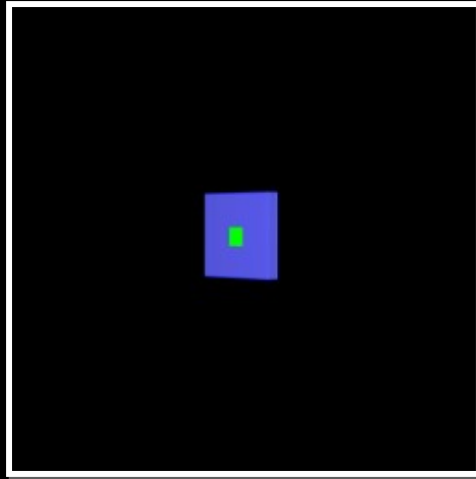
How Much to Refine?



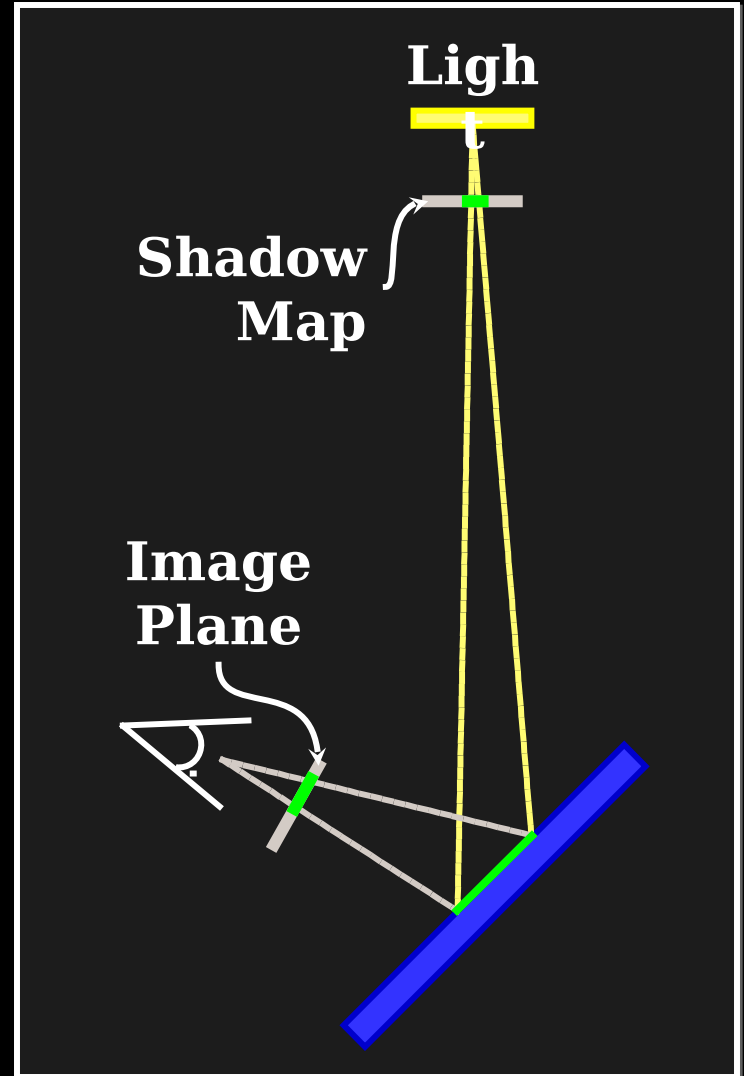
How Much to Refine?



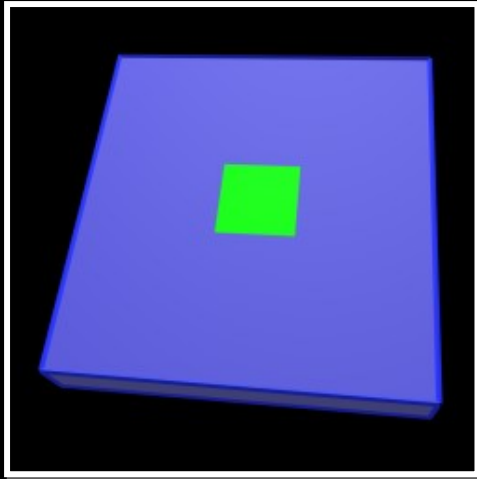
Eye View



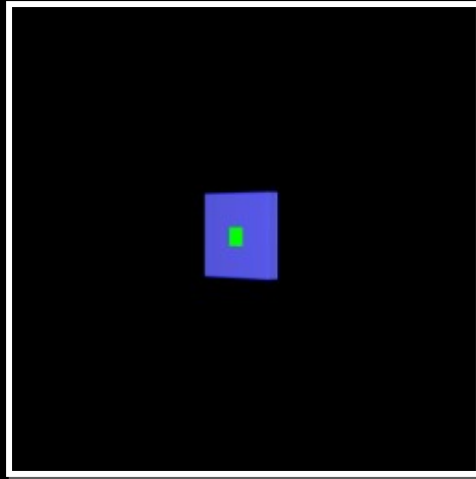
Shadow Map



How Much to Refine?



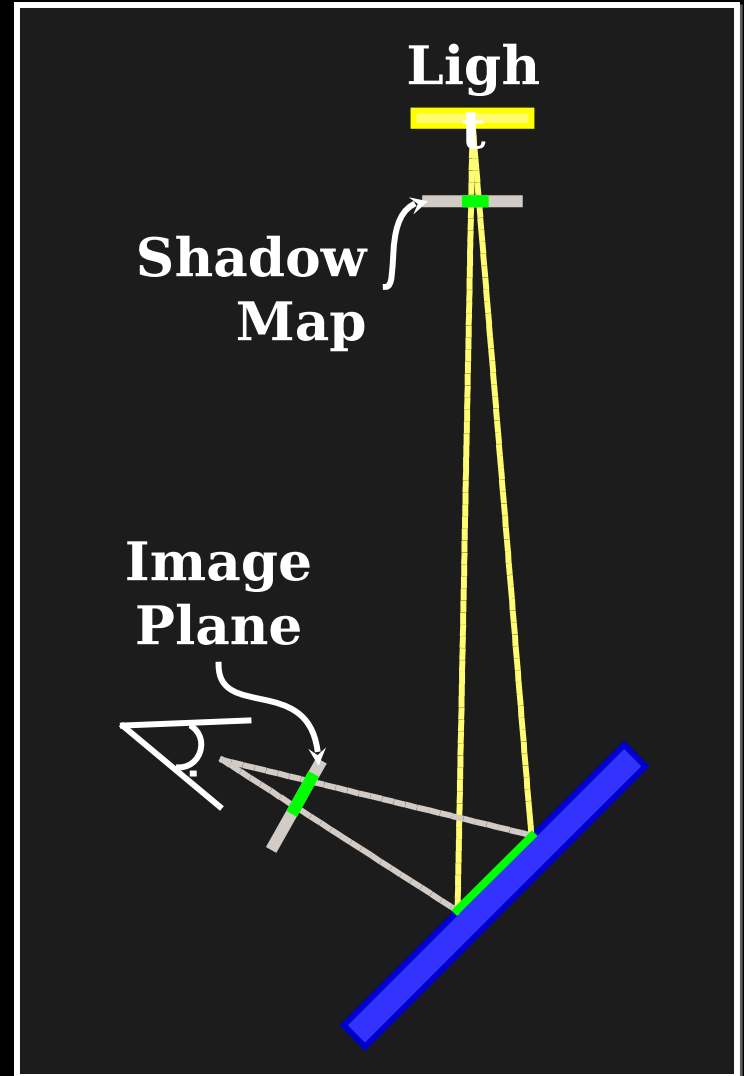
Eye View



Shadow Map

Goal:

Eye View
Projected Area = Shadow Map
Projected Area



Determining the Required Resolution

- For each transformed eye view pixel:

$N_{required}$ = Required Shadow Map Resolution

N_{eye} = Eye View Resolution

$$N_{required} = N_{eye} \cdot \frac{\text{Eye View Pixel Projected Area}}{\text{Shadow Map Pixel Projected Area}}$$

Refinement

- For each potential shadow map piece:
 - $Max N_{required} > N_{current} \rightarrow$ refinement needed
 - $Cost = a (Max N_{required}) + b$
 - $Benefit$ = number of resolution-mismatched edge pixels that would be resolved
- Update data structure

Determining the Projected Pixel Size

- Use hardware texture-mapping features
 - Mip-mapping
 - Trilinear / anisotropic filtering
- Create **uniform world texel size**
 - Compute projected size in eye view
 - Compute projected size in shadow map
 - Take ratio to determine if refinement is needed

Mip-map Specifics

- Use alpha channel for calculations
- Define white mip-mapped texture
- Encode texture size in alpha channel:

1
1 x 1
pixel
s

2 2
2 x 2
pixel
s

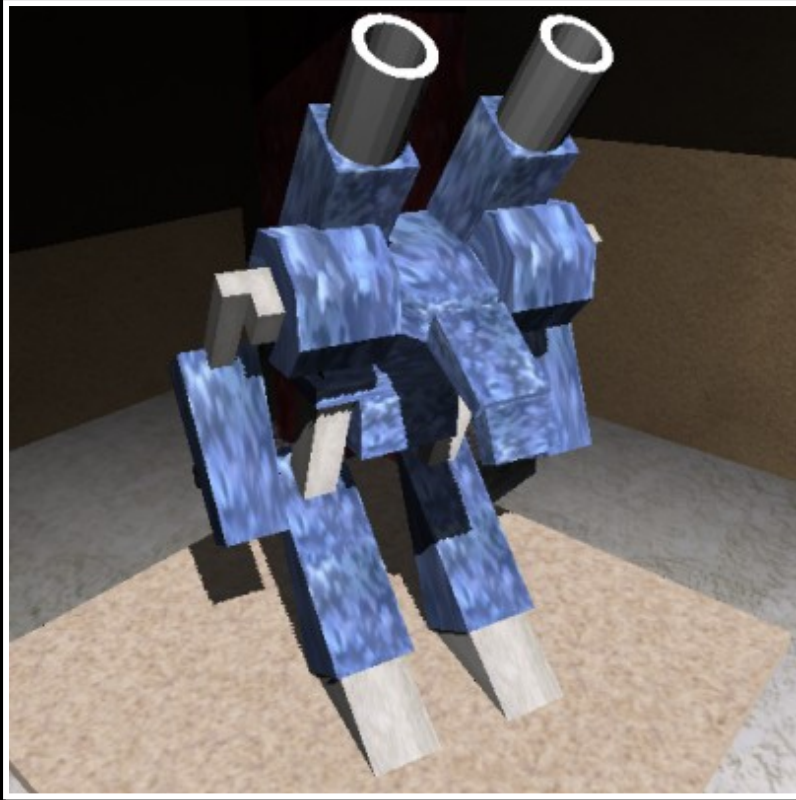
4 4 4 4
4 4 4 4
4 4 4 4
4 4 4 4
4 x 4
pixels

...

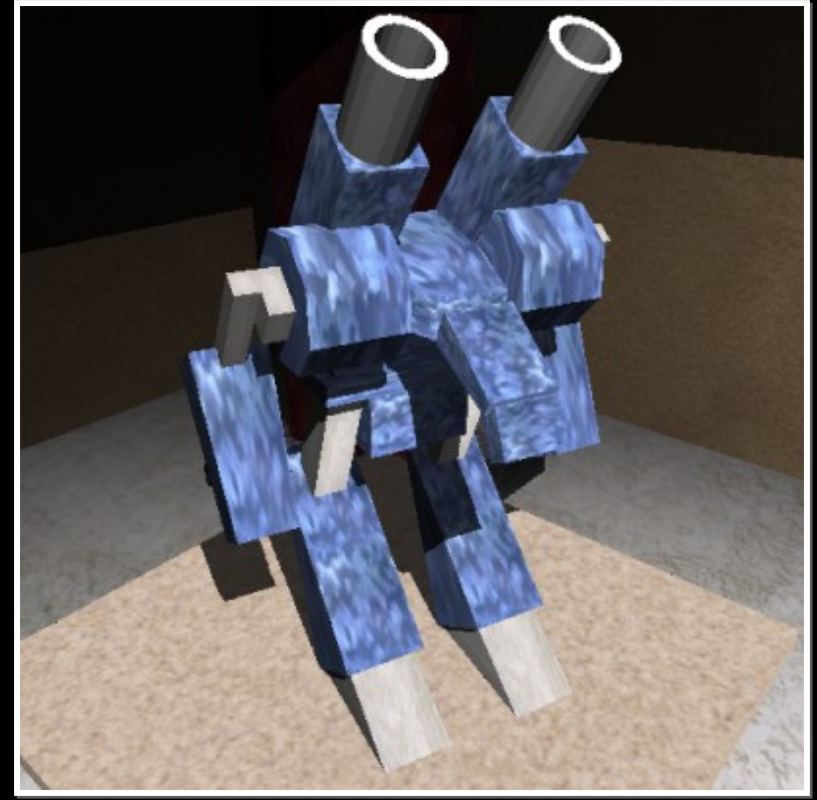
Mip-map Specifics (Cont'd)

- For each polygon, pick texture coordinates that create a uniform texel size (in world space)
- Modulate with polygon color
- Turn on trilinear interpolation, anisotropic filtering
- Render scene
- On read-back, alpha value has mip-map size (estimate of projected pixel size)
- **The graphics card does all the work!**

Results: Images (Robot)

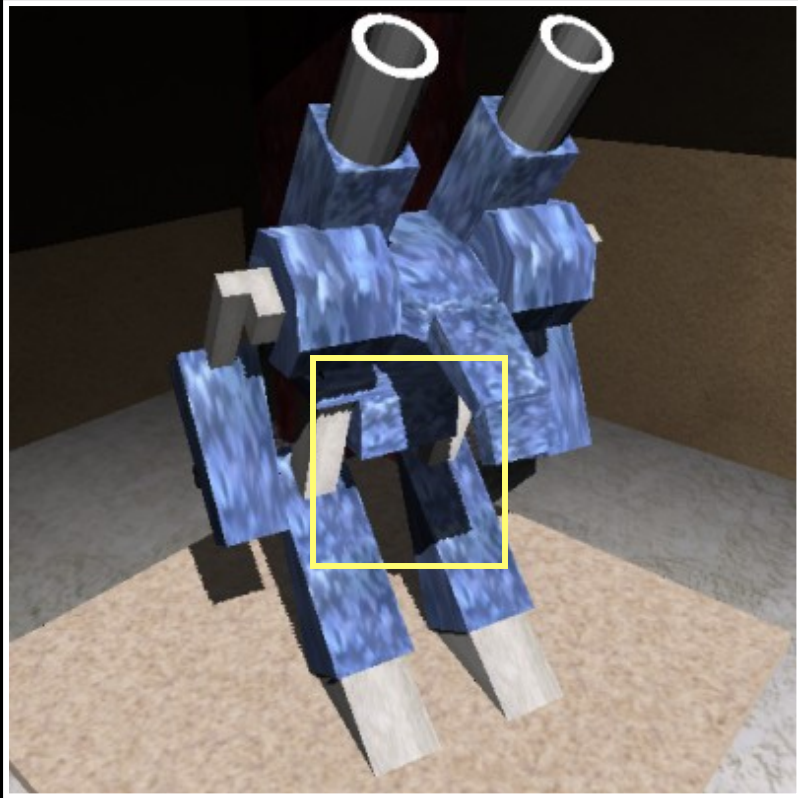


Conventional Shadow Map
(2048 x 2048 pixels)
16 MB Memory Usage

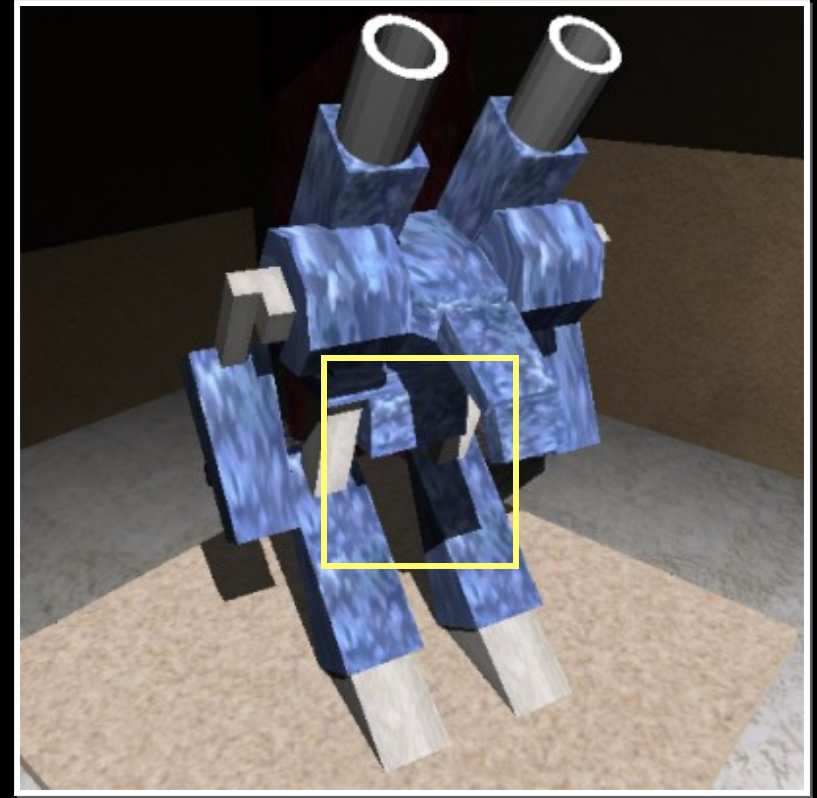


Adaptive Shadow Map
(Variable Resolution)
16 MB Memory

Results: Images (Robot)

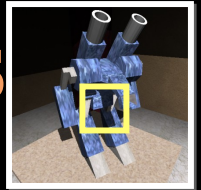


Conventional Shadow Map
(2048 x 2048 pixels)
16 MB Memory Usage



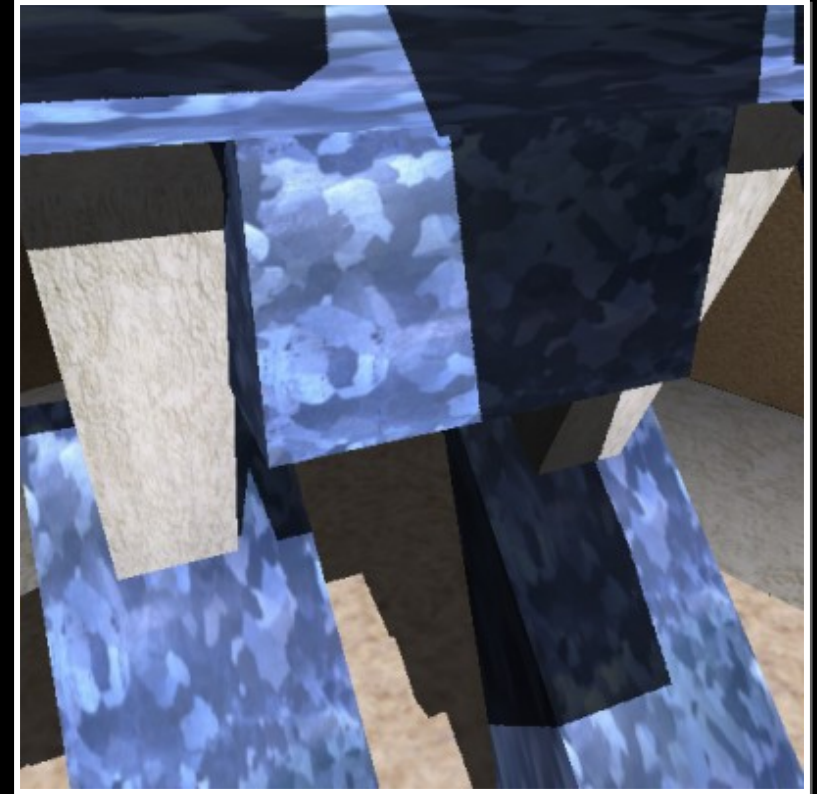
Adaptive Shadow Map
(Variable Resolution)
16 MB Memory

Results: Images (Robot Close Up)



Conventional Shadow Map

16 MB Memory Usage

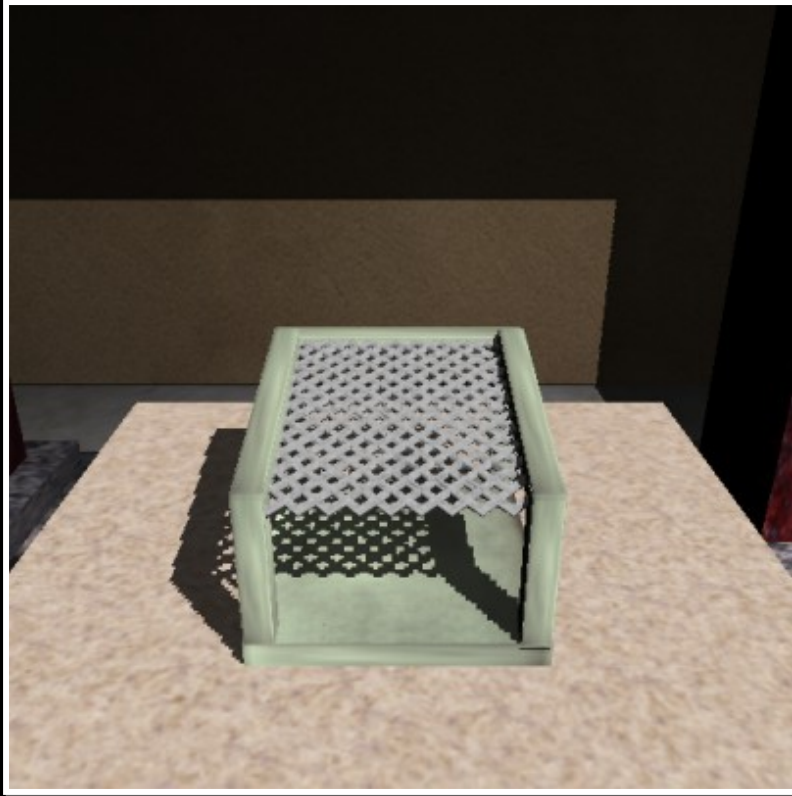


Adaptive Shadow Map

16 MB Memory Usage

Equivalent Conventional Shadow Map Size: $65,536 \times 65,536$ Pixels

Results: Images (Mesh)

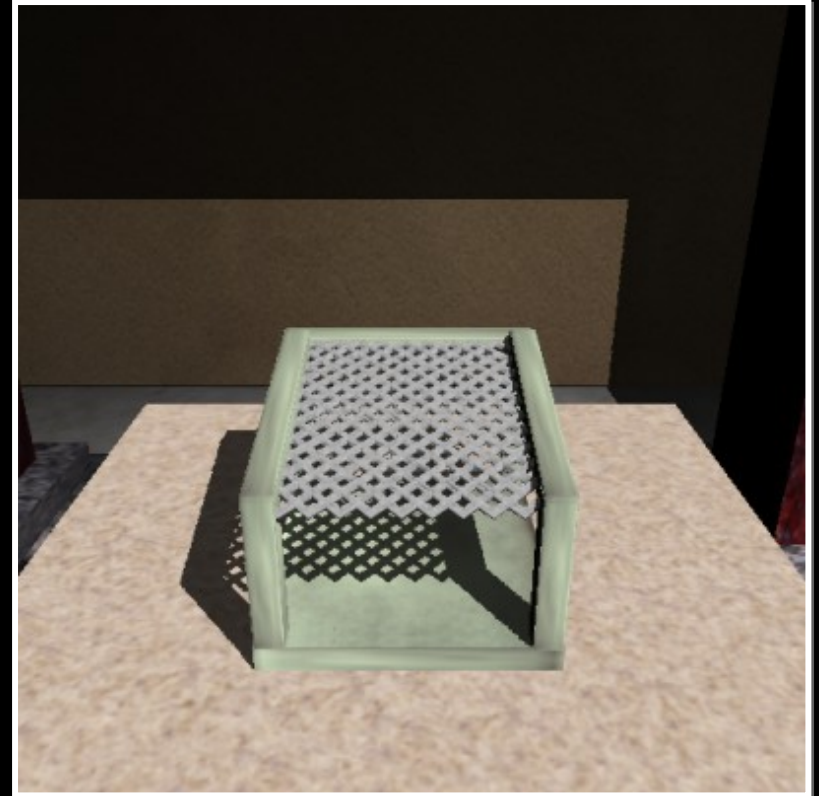


**Conventional Shadow
Map**

(2048 x 2048 pixels)

16 MB Memory

Usage



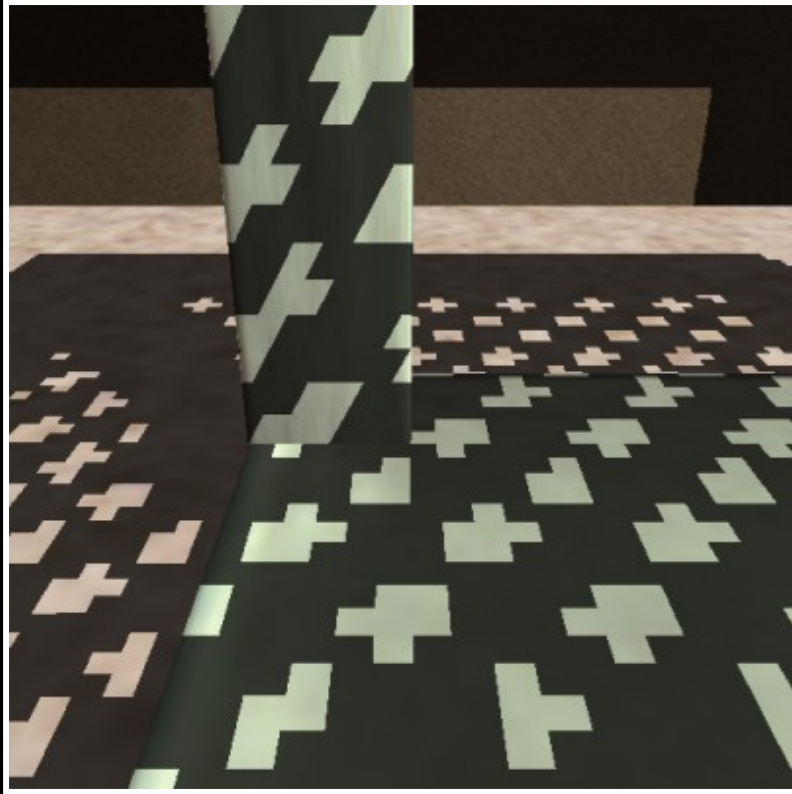
**Adaptive Shadow
Map**

**(Variable
Resolution)**

Resolution)

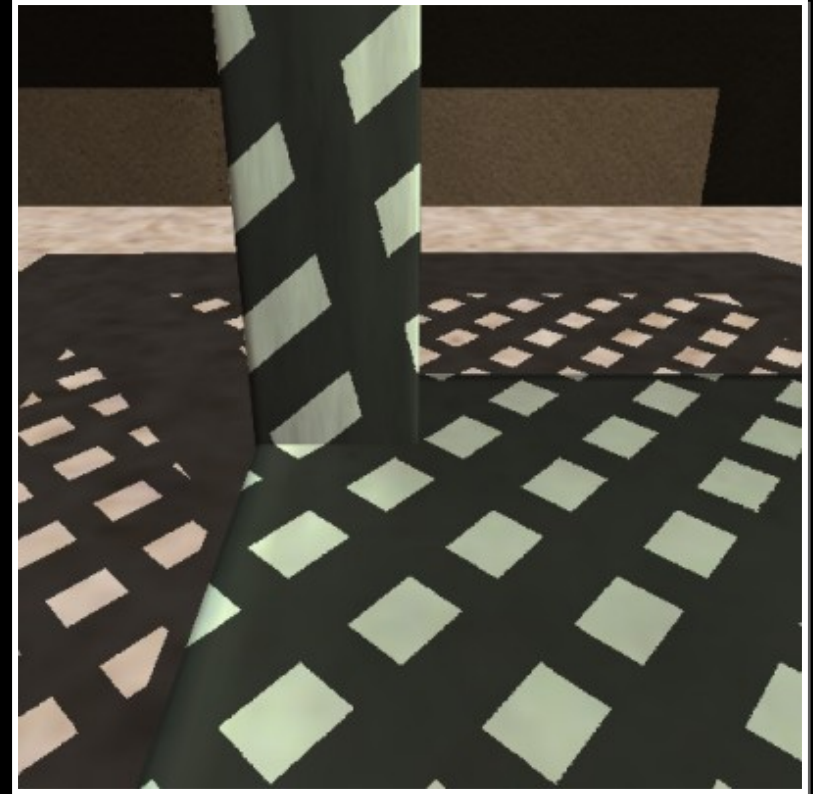
16 MB Memory

Results: Images (Mesh Close-Up)



Conventional Shadow Map

16 MB Memory Usage

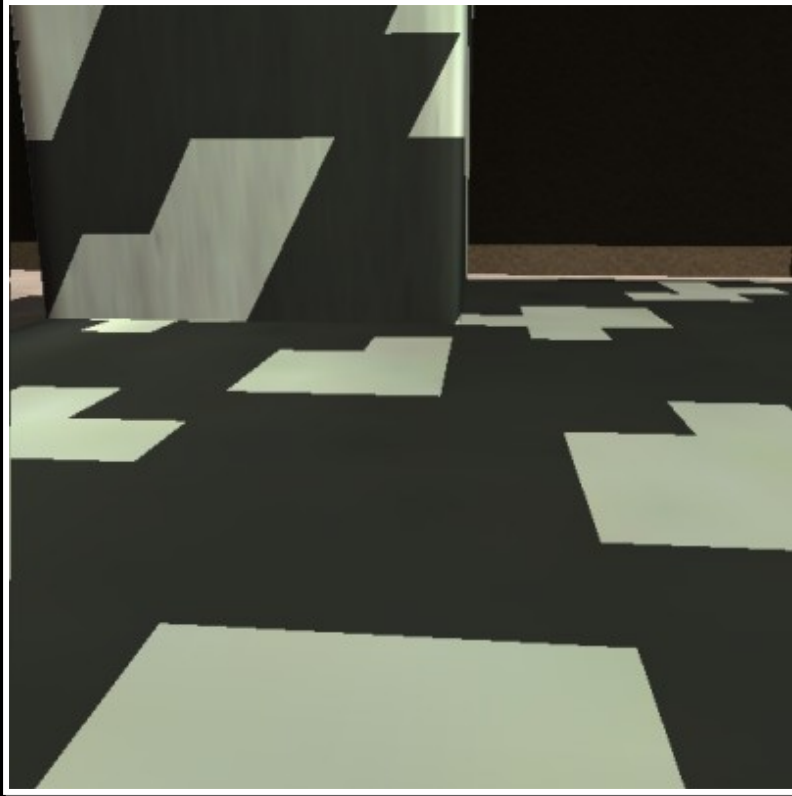


Adaptive Shadow Map

16 MB Memory Usage

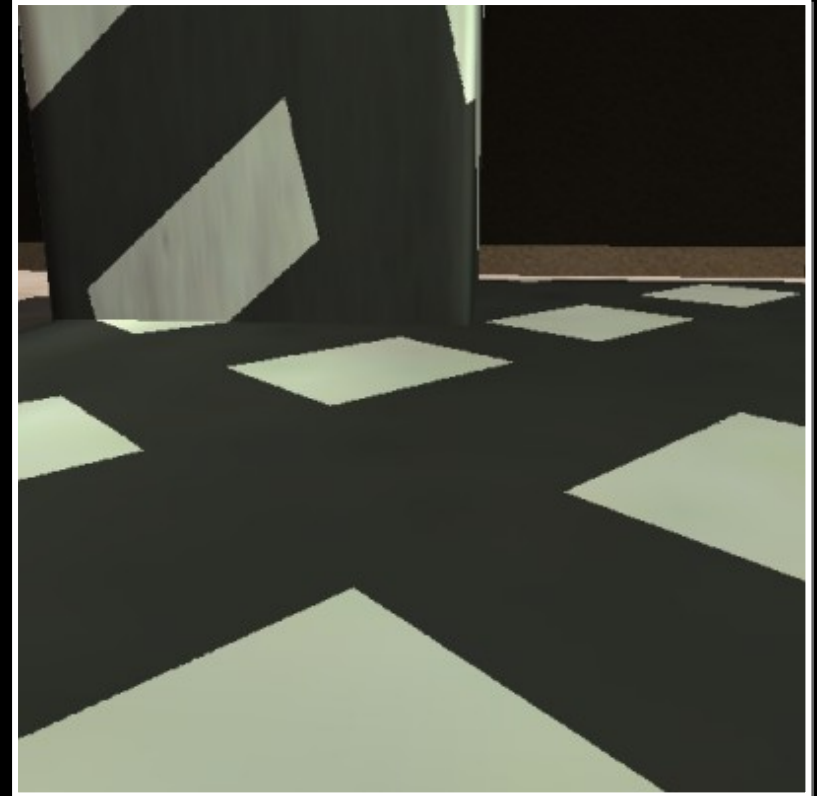
Equivalent Conventional Shadow Map Size: $65,536 \times 65,536$ Pixels

Results: Images (Mesh Close-Up)



Conventional Shadow Map

16 MB Memory Usage



Adaptive Shadow Map

16 MB Memory Usage

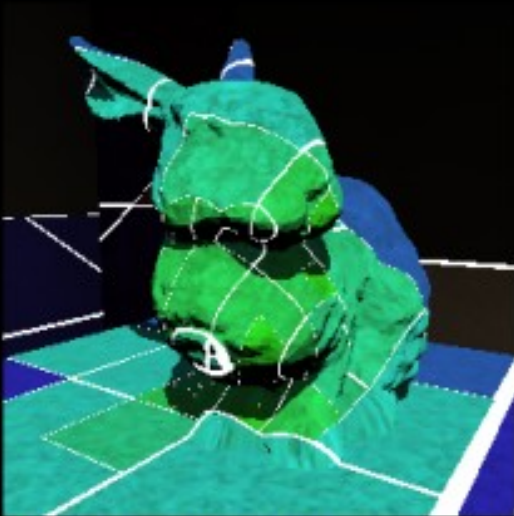
Equivalent Conventional Shadow Map Size: 524,288 × 524,288 Pixels

Results: Video

- Video filmed on a **1 GHz Pentium III** with an **NVIDIA GeForce2 Ultra** graphics board
- Adaptive Shadow Map is on the left

Results: Video (First Part)

- Visualization of Hierarchical Structure



ASM Memory Usage:
15.8 MB
Equivalent Shadow Map Size:
65536 x 65536 pixels
Frame Rate: ---

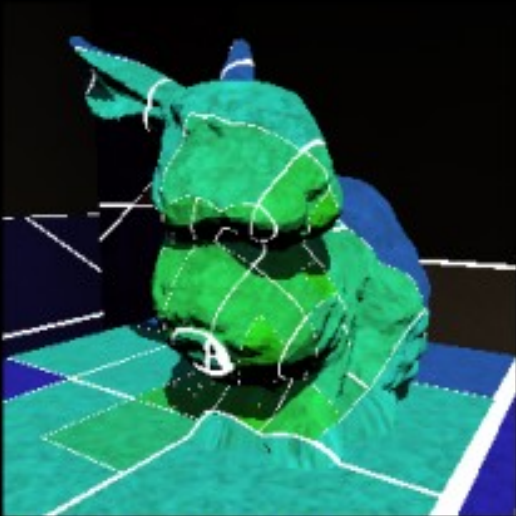
Adaptive Shadow Map



Grid Structure

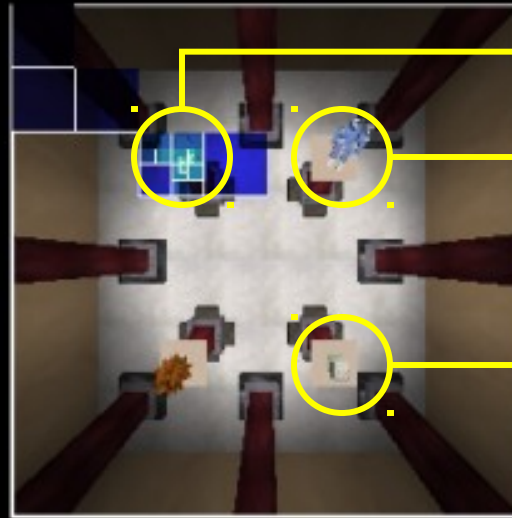
Results: Video (First Part)

- Visualization of Hierarchical Structure



ASM Memory Usage:
15.8 MB
Equivalent Shadow Map Size:
65536 x 65536 pixels
Frame Rate: ---

Adaptive Shadow Map



Bunny

Robo
t

Mes
h

Grid Structure

Future Work

- **Integration with other approaches:**
 - Deep shadow maps
 - Percentage closer filtering
 - Soft shadow techniques
 - Perceptual metrics

Conclusion

- **Adaptive Shadow Maps**
 - Address aliasing
 - Interactive rates
 - Hierarchical data structure
- **Features:**
 - View-driven
 - Constrained memory usage
 - Progressive
 - No manual intervention
 - Usage same as conventional shadow maps

Acknowledgements

- **Bruce Walter**
- **Eric Haines**
- **Parag Tole**
- **Fabio Pellacini**
- **Rich Levy**
- **Linda Stephenson**

Acknowledgements (Cont'd)

- **NVIDIA Corporation**
- **Intel Corporation**
- **National Science Foundation
Science and Technology Center for
Computer Graphics and Scientific
Visualization (ASC-8920219)**

Adaptive Shadow Maps

Questions/Comments?

Adaptive Shadow Maps

Randima Fernando
Sebastian Fernandez
Kavita Bala
Donald P. Greenberg